**AWS LAMBDA SECRET MANAGER**

Vamos a crear una función que tendrá acceso a Secrets Manager.

Devolveremos el objeto Keys Model con todas las claves para trabajar con este laboratorio.

Interfaz de usuario gráfica, Texto, Aplicación

Descripción generada automáticamente

<https://www.nuget.org/packages/HostInitActions>

<https://nodogmablog.bryanhogan.net/2022/04/accessing-aws-secrets-manager-from-net-lambda-functions-part-1-the-simple-way/>

Interfaz de usuario gráfica, Aplicación

Descripción generada automáticamente

Interfaz de usuario gráfica, Texto, Aplicación, Correo electrónico

Descripción generada automáticamente

{

"MySql": "server=awsmysqlpaco.ctinakxu7pmf.us-east-1.rds.amazonaws.com;database=personajesaws;user id=adminsql;password=Admin123",

"ApiPersonajes": "https://",

"S3Bucket:" : "",

"usermail": "micorreotestingcloudpgs@hotmail.com",

"passwordmail": "Hotmail12345",

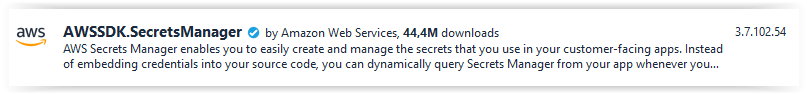
"host": "smtp.office365.com"

}

Interfaz de usuario gráfica, Texto, Aplicación, Correo electrónico

Descripción generada automáticamente

Una vez que tenemos los secretos, vamos a realizar una función Lambda para que nos devuelva los secretos en una clase JSON.



Interfaz de usuario gráfica, Texto, Aplicación, Word

Descripción generada automáticamente

**KEYSMODEL**

**public** **class** KeysModel

{

**public** **string** MySql { **get**; **set**; }

**public** **string** ApiPersonajes { **get**; **set**; }

**public** **string** S3Bucket { **get**; **set**; }

**public** **string** UserMail { **get**; **set**; }

**public** **string** PasswordMail { **get**; **set**; }

**public** **string** Host { **get**; **set**; }

}

**HELPERSECRETMANAGER**

**public** **static** **class** HelperSecretsManager

{

**public** **static** **async** Task<**string**> GetSecretsAsync()

    {

**string** secretName **=** "secretoslabs";

**string** region **=** "us-east-1";

        IAmazonSecretsManager client **=**

**new** AmazonSecretsManagerClient(RegionEndpoint**.**GetBySystemName(region));

        GetSecretValueRequest request **=** **new** GetSecretValueRequest

        {

            SecretId **=** secretName,

            VersionStage **=** "AWSCURRENT", // VersionStage defaults to AWSCURRENT if unspecified.

        };

        GetSecretValueResponse response;

        response **=** **await** client**.**GetSecretValueAsync(request);

**string** secret **=** response**.**SecretString;

**return** secret;

    }

}

Comprobamos el funcionamiento en el Api con **aws-configure** y vamos a visualizarlo en un Lambda

Interfaz de usuario gráfica, Aplicación

Descripción generada automáticamente

Interfaz de usuario gráfica, Texto, Aplicación

Descripción generada automáticamente

Interfaz de usuario gráfica, Texto, Aplicación, Word

Descripción generada automáticamente

**FUNCTION**

**public** KeysModel **FunctionHandler**(**string** input, ILambdaContext context)

{

**string** data **=**

        HelperSecretsManager**.**GetSecretsAsync()**.**GetAwaiter()**.**GetResult();

    KeysModel model **=** JsonConvert**.**DeserializeObject<KeysModel>(data);

**return** model;

}

Creamos un Role en IAM para poder ejecutar la función.

Interfaz de usuario gráfica, Texto, Aplicación, Word

Descripción generada automáticamente

Abrimos el fichero dotnet y ponemos nuestro nuevo role dentro del fichero

"function-role": "role-execution-lambda"

Publicamos nuestra función en AWS como Lambda con la siguiente instrucción

**dotnet lambda deploy-function lambda-secrets**

Incluimos un Trigger de tipo Api Gateway y enlazamos la función. Probamos su funcionalidad

Interfaz de usuario gráfica, Texto, Aplicación

Descripción generada automáticamente

Necesitamos saber el ARN de **Secrets Manager** para darle permisos a la función sobre él.

Interfaz de usuario gráfica, Texto, Aplicación, Correo electrónico, Teams

Descripción generada automáticamente

Vamos a dar secretos solamente a esta función para que pueda leer únicamente

nuestro Secret Manager personalizado, ninguno más.

Abrimos nuestro Role de IAM

Interfaz de usuario gráfica, Texto, Aplicación, Correo electrónico

Descripción generada automáticamente

Interfaz de usuario gráfica, Texto, Aplicación, Correo electrónico

Descripción generada automáticamente

Interfaz de usuario gráfica, Texto, Aplicación, Correo electrónico

Descripción generada automáticamente

Interfaz de usuario gráfica, Texto, Aplicación

Descripción generada automáticamente

Interfaz de usuario gráfica, Aplicación

Descripción generada automáticamente

Interfaz de usuario gráfica, Texto, Aplicación, Correo electrónico

Descripción generada automáticamente

Volvemos a probar nuestra función con Api Gateway y podremos comprobar su funcionalidad correcta

Interfaz de usuario gráfica, Texto, Aplicación, Correo electrónico

Descripción generada automáticamente

Ponemos nuestra Api Gateway en producción

Interfaz de usuario gráfica, Texto, Aplicación

Descripción generada automáticamente

Interfaz de usuario gráfica, Texto, Aplicación

Descripción generada automáticamente

Interfaz de usuario gráfica, Texto, Aplicación, Correo electrónico

Descripción generada automáticamente